39

TELEDESIC:
A PRODUCT, PROCESS AND SUPPLY CHAIN DESIGN METHODOLOGY

by

Lance Clifford Mansfield

B.S., Electrical Engineering, Seattle Pacific University, 1993

Submitted to the Department of Electrical Engineering and Computer Science
and the Sloan School of Management
in Partial Fulfillment of the Requirements for the Degrees of

Master of Science in Electrical Engineering
And
Master of Science in Business Administration

in Conjunction with the
Leaders for Manufacturing Program

at the
Massachusetts Institute of Technology
June 1998

Signature of Author_____
Department of Electrical Engineering and Computer Science
Sloan School of Management
May 8, 1998

Certified by_____
Professor Edward Crawley, Thesis Advisor
Department of Aeronautics and Astronautics

Certified by_____
Professor Charles Fine, Thesis Advisor
Sloan School of Management

Accepted by_____
Arthur C. Smith, Chairman, Committee on Graduate Studies
Department of Electrical Engineering and Computer Science

Accepted by_____
Larry Abeln, Director of Master's Program
Sloan School of Management

TELEDESIC:
A PRODUCT, PROCESS AND SUPPLY CHAIN DESIGN METHODOLOGY

by

Lance Clifford Mansfield

B.S., Electrical Engineering, Seattle Pacific University, 1993

Submitted to the Department of Electrical Engineering and Computer Science
and the Sloan School of Management
in Partial Fulfillment of the Requirements for the Degrees of

Master of Science in Electrical Engineering
And
Master of Science in Business Administration

Abstract

Current industry best practice in design achieves concurrent engineering through Integrated Product Teams (IPTs) at the design phase of a project. The Boeing Company's Teledesic project provides the opportunity to integrate the product, process and supply chain designs during the concept phase of the project, where the design leverage is the greatest. The difficult functional, cost and schedule requirements on the Teledesic project make **concurrent system engineering** a key enabler for project success.

To achieve this concurrency, Boeing's core competency in product system engineering is expanded to include the process and supply chain architectures. This expansion will provide the ability to concretely and quantitatively trade requirements across all three architectures. This work will outline Boeing's traditional system engineering methodology and how the Teledesic project expanded this methodology to allow for optimization across product, process and supply chain boundaries. The implementation strategy for this expansion is also presented. Two project examples communicate the effectiveness of the expanded methodology in trading system requirements across architecture boundaries.

Thesis Supervisors:

Edward Crawley
Professor of Aeronautics and Astronautics

Charles Fine
Professor of Management

Table of Contents

Table of Figures

Acknowledgements

First, I'd like to acknowledge my Boeing and Teledesic supervisors, Mark Ellis and Jim Miller, for creating space on the Teledesic team for me and paving the way for a very successful internship. You both have taught me, through your words and actions, how leaders communicate a vision, assemble the right team and then give them the freedom to implement. I am looking forward to working for you again one day.

Likewise, I'd like to thank Boeing's Teledesic DefineProduce Team, a group of skilled and experienced managers who tackled the Teledesic production system with the enthusiasm of new hires and the wisdom of savvy veterans. In particular, I'd like to thank:

- Eric Jacobs, for showing me that ship building is as object-oriented as software and that cost is the mother of all requirements.
- Norm Beougher, for giving me my Kent and Boeing bearings, including my new favorite Thai place.
- Joe Fletcher, for laughing with me as the loner and the intern kept down the fort.
- Ken Coe, for combining energy and intelligence in logistics, of all places.
- Courtney Weiss, for extending me a listening ear and Texas hospitality.
- Paul Hesla, for actually quantifying a day of flow time, to the amazement of the design engineers.
- And Sam Coffield and Rich Munroe, for showing me why I, too, should love flying and motorbikes.

I'd also like to thank my MIT thesis supervisors, Charley Fine and Ed Crawley. They have provided me with a strong sense of the academic history behind new product development and systems engineering, loaned me their frameworks for a test drive and helped me understand the unique opportunity Teledesic represents. Likewise, I'd like to acknowledge the Leaders for Manufacturing program, for providing me the education, structure and space to carry out this endeavor.

Lastly but most importantly, I'd like to thank my wife Abby, who left friends and family in Seattle and ventured out to Cambridge to help me live out a life-long dream. Thank-you for loving and supporting me throughout our East Coast adventure together.

Disclaimer

The details about the Teledesic project outlined below have been stylized to match information that is currently available in the public domain. Because the author had access to the actual requirements and system architectures, many of details outlined below are known to be in error. In addition, the design has been modified since the completion of the research, making the author's knowledge incomplete. The examples from the Teledesic project are meant to illustrate how a concurrent system engineering methodology would be implemented on a project like Teledesic, not to communicate any specific design information. However, the description of the methodology and the results as demonstrated on the Teledesic project are factual and meant to show the effectiveness of the methodology in addressing complex product, process and supply chain architecture challenges.

1 Introduction

Current industry best practice in design achieves concurrent engineering through Integrated Product Teams (IPTs) at the design phase of a project. [1] However, Boeing's Teledesic project provides the opportunity to integrate the product, process and supply chain designs during the concept phase of the project, where the design leverage is the greatest. [2] The difficult functional, cost and schedule requirements in the process and supply chain architectures makes concurrent system engineering a key enabler for project success.

1.1 Current Best Practice in Product Design

Current best practice in product design achieves concurrent engineering through the use of IPTs. Having the designers, manufacturing engineers and, increasingly, the major suppliers participate on a cross-functional design team is the best way to ensure a design optimally meets the requirements. [1] However, other work has shown that the majority of project costs are determined in the concept phase of a project, before IPTs are traditionally engaged. [2]

In addition, most tools for production and supply chain system engineering focus in the improvement of existing facilities or processes, not on the creation of new systems. Lean Manufacturing *reduces* waste and flow time. [3] Theory of Constraints *maximizes* throughput. [4] Total Quality Management *reduces* variation. [5] Linear Programming *optimizes* key output parameters. [6] Each of these tools improves existing operations.

One of the few tools for designing new production and supply chain systems, Intel's Copy Exactly, tries to duplicate the performance of the best existing integrated circuit fabrication facility in the new plant. [7] While effective in Intel's context, simply duplicating today's best satellite production and supply chain system fails to meet the Teledesic project objectives. This absence of tools to accomplish the production and supply chain system engineering task makes Boeing's chosen methodology a benchmark for others attempting significant innovation in the production and supply chain system design.

1.2 The Teledesic Challenge

Like many companies with a strong new product development heritage, The Boeing Company has traditionally integrated the product and process design activities through **Integrated Product Teams (IPTs)**. The Teledesic project provides the opportunity to take this integration to the next level and pursue a concurrent **system engineering** effort in the product, process and supply chain architectures.

The challenges for the Teledesic production and supply chain system design are significant. No mass manufacturing production systems exist that can produce the rate of satellites needed for the success of Teledesic, as illustrated in Figure 1. Also, radically reduced final assembly flow time is required from the supply chain, as illustrated in Figure 2. [8] Clearly, the Teledesic project requires major innovation in the process and supply chain architectures.

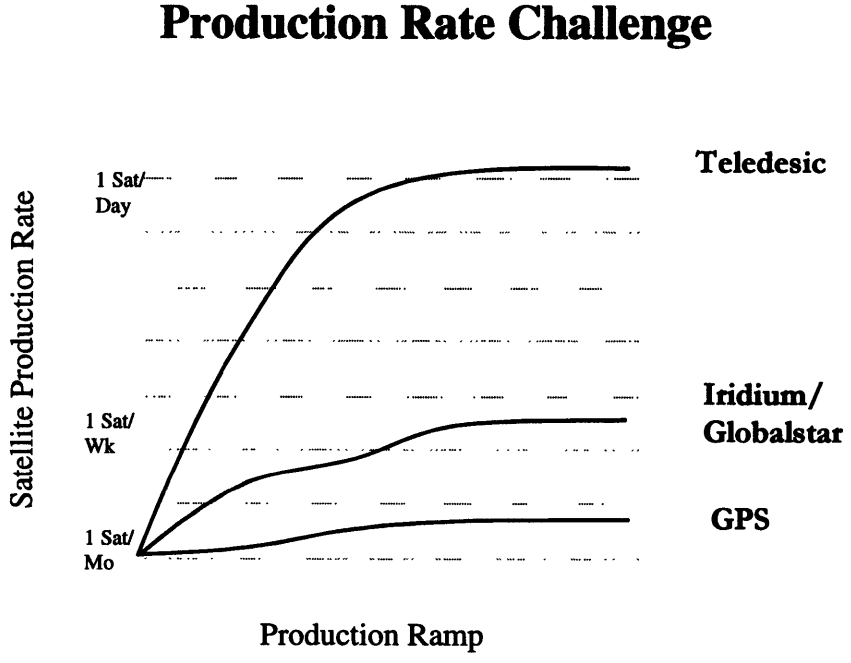

Figure 1: Production Rate Challenge

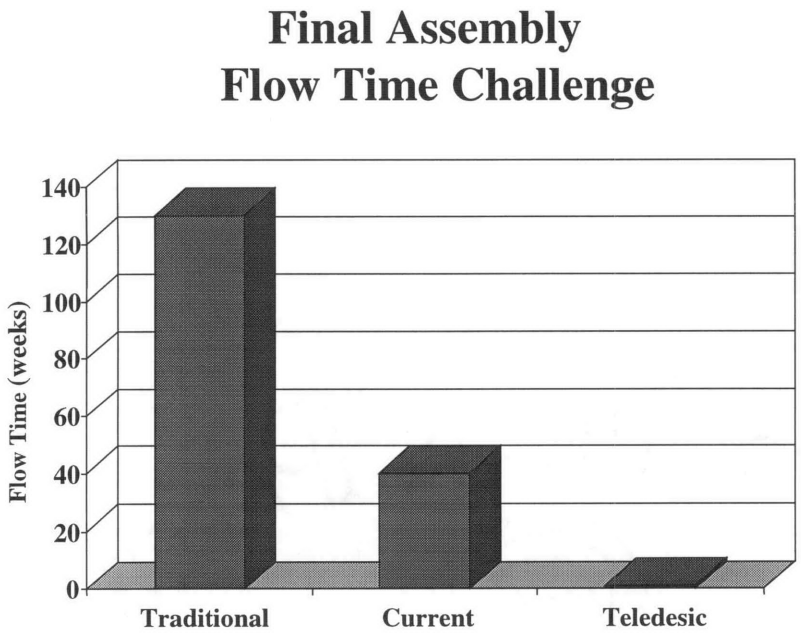

Figure 2: Final Assembly Flow Time Challenge

In order to meet these difficult project requirements, a concurrent system engineering process is necessary. This methodology must concretely and quantitatively trade requirements across product, process and supply chain boundaries. This work outlines how Boeing expanded its core competence in product system engineering to the process and supply chain architectures, enabling concurrent system engineering in the concept phase of the project.

1.3 Framework Correlation to the Methodology Recommendations

The implemented system engineering expansion is strongly correlated to frameworks recently developed at the Massachusetts Institute of Technology (MIT) for analyzing the effectiveness of system designs. An outline of these frameworks will serve as a useful backdrop for the subsequent discussion of the system engineering methodology modifications.

One such framework is the Fine/Cohen FAT 3-D Matrix [9], which draws attention to the importance of including supply chain architecture as a third component in system engineering, as illustrated in Figure 3. Specifically, the Focus, Architecture, Technology, Product, Process and Supply Chain linkages show the

dependencies between the architectures. If these linkages to the supply chain architecture are ignored, then the overall system integrity will be reduced and the ability of the architecture to meet its objectives will be at risk.

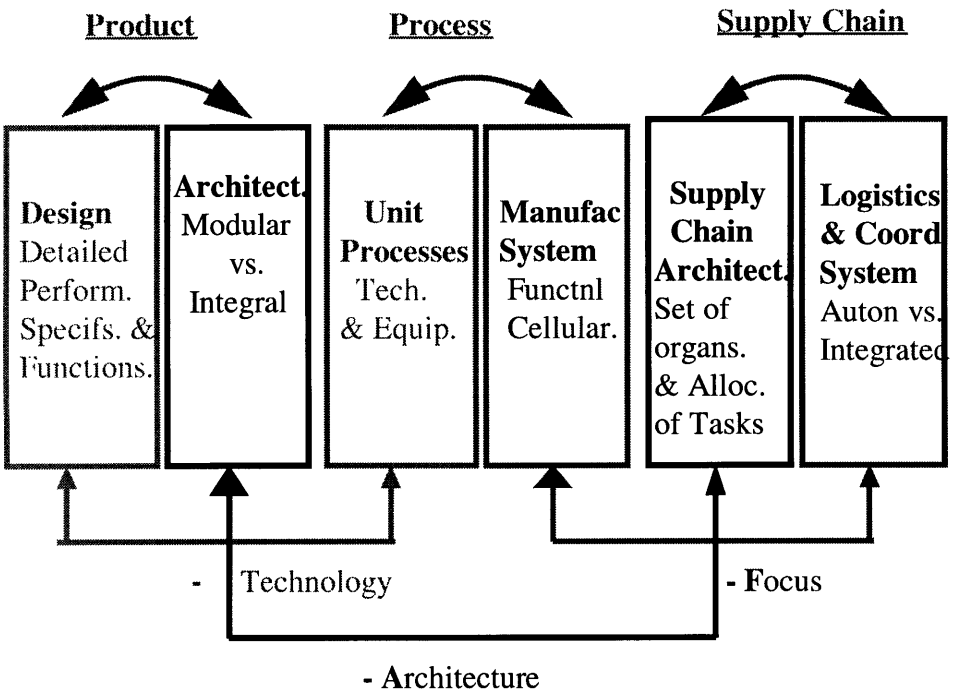

Figure 3: Fine/Cohen FAT 3-D Matrix

The recent success of several firms in competing on supply chain efficiencies also highlights the imperative of explicitly designing the supply chain. Dell Computer, Amazon.com and L.L Bean are three examples of organizations that use supply chain architecture and logistics to be a chief source of competitive advantage. Several other companies, notably Chrysler, are re-aligning their product and process design systems to maximize their strength in supply chain architecture design. [9] Analyzing the markets where these companies compete with the FAT 3-D Matrix highlights the mismatches between architectures in the current system, showing why these new entrants can compete successfully against the bigger and stronger incumbents. By explicitly designing the supply chain to be in harmony with the product and process architectures, the overall integrity of the products and systems created by the firm can be enhanced.

Another important new framework, the Crawley Total Holistic View of Product/Process Architecture [10], emphasizes the importance of explicit process and operational design, as illustrated in Figure 4. The black dots represent the activities that are traditionally included in a new project effort. The product goals are defined, the function of the subsystems is specified in the form of requirements, those requirements are allocated to products and lastly, the execution team is specified.

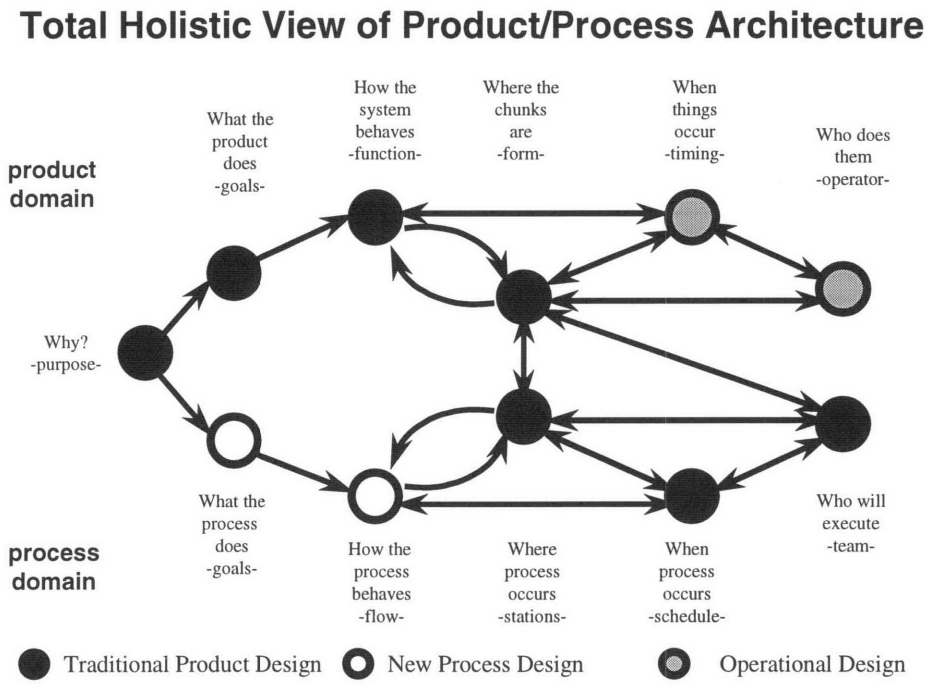

Figure 4: Crawley Total Holistic View of Product/Process Architecture

The white dots illustrate that the often forgotten process design activities strongly parallel the product design, and that, indeed, new processes can be designed like new products. Especially for Teledesic, where the current *industry* process capability does not meet the project requirements, explicit design of the processes is essential to project completion. These linkages also highlight the opportunity for the development of process design tools that address the new process design challenge in a strong and direct way.

The gray dots outline the operational design segment. Especially for the Teledesic Network, where the system will be constantly managed throughout it's lifetime, the explicit design of the operational processes and procedures is central to the successful completion of the project requirements.

These two frameworks help to highlight three often overlooked elements of the system engineering process: supply chain, process and operational design. The explicit design of these elements will strongly enhance a traditional product-centric system engineering process.

2 Teledesic Project Context

The Teledesic project is characterized by it's difficult functional, cost and schedule requirements, which drive the need to innovate concurrently in the product, process and supply chain architectures. To address the production and supply chain system design challenges, a new team, the DefineProduce team, was created within the traditionally product-oriented System Engineering group. The unique Teledesic project requirements and the corresponding unique organizational structure provide the backdrop for the expansion of the requirement-centric product system engineering methodology to include the process and supply chain architectures.

2.1 The Teledesic Network System Requirements

The Teledesic program is defined by difficult functional, cost and schedule requirements. In addition, the functional, cost and schedule requirements are considered equally important. This equal weighing differs from previous satellite systems, where the functional requirements were more important than the cost and schedule requirements. The increased importance of cost and schedule attracts resources and visibility to the process and supply chain design activities. This visibility enables changes to the traditional process and supply chain design methodology that would not normally be feasible. The stringent Teledesic system requirements are outlined below.

Functional Requirements

In the words of the Teledesic Corporation, the Teledesic Network is a high-capacity broadband network that will provide affordable access to interactive broadband communication to all areas of the Earth, including those areas that cannot be served economically by any other means. [11]

This mission translates into a network with global coverage, "Internet-like" flexibility and robustness, "fiber-like" Quality of Service (QOS), capacity for millions of simultaneous users and compatibility with applications that are based on today and tomorrow's protocols. "Fiber-like" QOS means the system will have fiber-like performance in the dimensions of latency, error rate, service availability and data

throughput. [11] These goals pose very stringent requirements to the satellite design, making the Teledesic Network the most complex satellite system ever attempted.

Schedule Requirements

As described by the Teledesic Corporation, the target schedule is:

- 1997 FCC license granted, Boeing hired as prime contractor.
- 1999 Start pre-production activities.
- 2000 Full-scale production.
- 2001 First satellite launch.
- 2002 Commercial service begins. [11]

For Boeing, this translates into a project cycle of 5 years, broken up as follows:

System design:	1 year
Detailed design/Verification:	2 years
Production and Launch:	2 years

For a system as complex as the Teledesic Network, this schedule is very aggressive. For comparison, it takes the Hughes Corporation 18 months to design and deploy one GEO satellite, using a baseline design and a dedicated launch. To deploy a system of 300 satellites with an extended launch campaign under these schedule constraints will demand very efficient execution of the project plan.

Cost Requirements

Again in the estimates of the Teledesic Corporation, the design, production and deployment of the 288 satellites in the Teledesic Network will cost approximately $9 billion. [11] To put this cost in perspective Iridium, the world's largest pre-Teledesic satellite project, paid Motorola $3.4 billion for 72 LEO satellites, leading to a cost of approximately $47 million per satellite. [12] A typical GEO satellite mission typically costs in the $125 million to $175 million range. [13] The Teledesic Network technical requirements are much greater than the Iridium system, and yet the average cost of a Teledesic satellite will be on the same order of magnitude as an Iridium satellite.

The Teledesic Network requirements impose constraints that are much more rigorous than other satellite projects. In order to meet these requirements, the product, process and supply chain architectures must be globally optimized.

2.2 Teledesic Project Organizational Framework

To address the challenges in the production and supply chain architectures, a new team, the DefineProduce team, was added to Boeing's traditional organizational framework. This framework consists of project management, systems engineering, IPTs, production functions and production centers. Each team's responsibilities and deliverables are outlined below.

Team	Responsibilities	Deliverables
Project Management	Program definition Program monitoring	Cost budget Program schedule Risk identification and mitigation process Make/buy decisions Strategic supplier selection
System Engineering (including the DefineProduce team)	Decompose program functional, cost and schedule requirements	IPT requirement documents IPT interface documents
IPTs	Design products	Product definition SOW definition Supplier selection
Production Functions	Design production system	Program-level functional plan
Production Centers	Produce products	Product and documentation

Figure 5: Teledesic Organizational Responsibilities

Project Management

The project management is responsible for the identification of the proper personnel for each of the various teams, and for determining the make/buy plan for the program. The project management sets the program level goals, as well as strongly influences the global architecture decisions. In order to accomplish these tasks, groups responsible for cost accounting, scheduling and risk identification and mitigation support the project management effort.

System Engineering

The System Engineering teams are responsible for decomposing the program-level requirements into the complete but minimal set of requirements for every major element in the architecture. This activity includes defining the top levels of the product architecture. They also define the interfaces between the architecture elements, producing interface control documents that define these boundaries.

DefineProduce Team

Because of the increased visibility of the process and supply chain architectures on the Teledesic project, a new system engineering team was created to specifically address these architecture activities. This team had the responsibility to:

- define the methodology for the process and supply chain architecture development
- develop the high level process architecture
- develop the high level supply chain architecture
- determine the key process requirements for the satellite architecture
- define the process requirements for the production centers
- define the requirements for the supply chain architecture teams
- achieve acceptance of the requirements from the IPTs

The DefineProduce team is made up of managers from Boeing's traditional production functions. This research was completed within this team's charter.

Integrated Product Teams

The IPTs deliver a product which implements the requirements defined by the system engineers and the plans defined by the production functions. They manage both the design and the production of their product. They define and verify requirements for the production centers. Because of they have both product and process responsibility, these teams typically have members with cross-functional experience, as well a few members with deep knowledge of the necessary technologies and process capabilities. In fact, Boeing is one of the leaders in integrating manufacturing into the design process through their creation

of design-build teams almost 17 years ago. The IPTs are the traditional place that the product and process issues are surfaced and resolved.

Production Functions

Production functions, including the Quality, Manufacturing, Logistics, Parts, Materials and Process (PM&P), System Test, Materiel and Digitally Driven Enterprise (DDE) groups, coordinate the important production architecture activities.

While the production centers are selected by the IPTs and given the ability to control activity within their plants, the various functions usually coordinate program-level activity throughout the supply chain, mandating certain activities in order to optimize the production system. These program-level systems and methodologies are usually documented in plans, and those plans are often referenced in legal documents between the program and the supplier network. This plan-centric system is the historical communication medium of the program functional expectations to all affected production centers.

Production Centers

Production centers produce the products defined by the IPT designs, following the processes outlined by the IPT Statement of Work (SOW). They also help the IPTs obtain optimal product and process definitions by communicating their current production process capability.

3 Boeing's Traditional System Engineering Methodology

Boeing's traditional system engineering methodology is, in fact, two distinct methodologies, as outlined in Figure 6. The product architecture is implemented using a requirement-centric methodology, where the functional requirements are systematically decomposed and rationalized to the architecture. Conflicts between the architecture and the requirements are reconciled in a well-defined process, utilizing trade studies to measure the relative value of the lower requirements in terms of program goals.

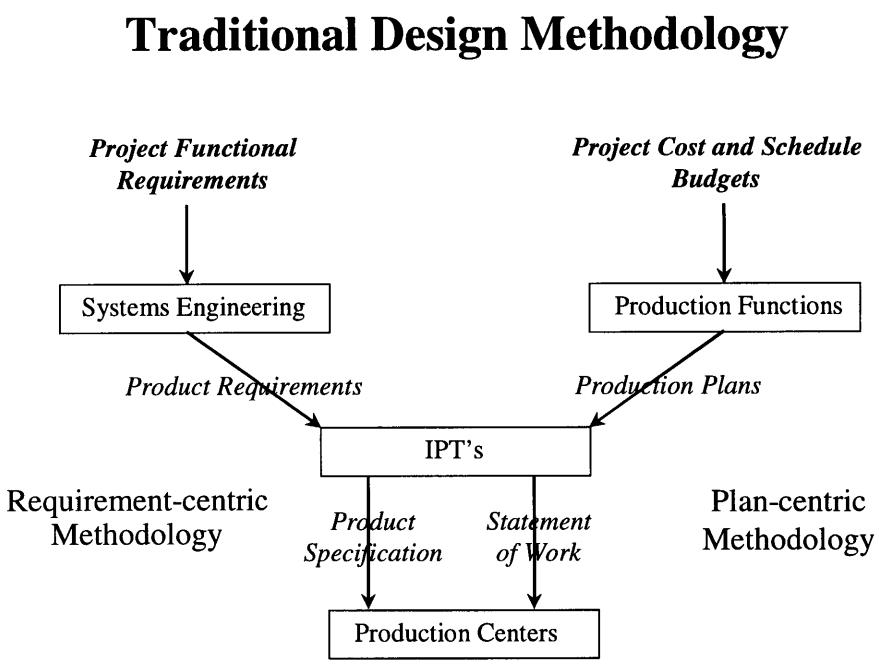

Figure 6: Traditional System Engineering Methodology

On the other hand, the process architecture relies on a plan-centric methodology. The cost and schedule goals are allocated to the various production functions, which develop plans to show how they are going to meet their targets. These plans are then negotiated with the IPTs, who develop a SOW for their designed product. This SOW becomes the core of the process requirement contract with the production centers.

The supply chain architecture is not explicitly designed, but is the result of several process architecture design activities. The implementation of the supply chain architecture is largely through the execution of

the production functions. Detailed explanations of the system engineering methodologies and their resulting interactions in the IPT design process are outlined below.

3.1 Traditional Product System Engineering Methodology

Boeing's traditional product system engineering methodology is centered on requirements. The system engineering group begins by taking the program level functional requirements and decomposing them, as in Figure 7. Typically, these high level requirements are captured in a requirement document that becomes part of the legal contract with the customer. Because of the legal nature of this document and the traditional costs associated with changing it, significant effort ensures that the requirements are consistent, clearly defined and do not overly constrain the design.

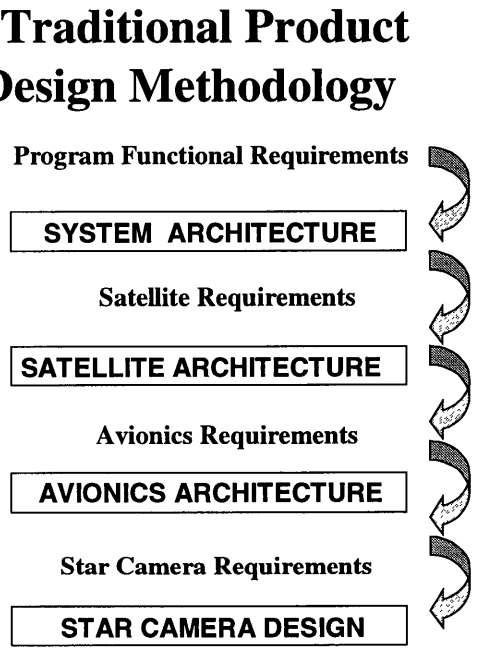

Figure 7: Traditional Product System Engineering Methodology

Concurrently, the system engineers are also developing potential system architecture solutions. Through design trades, they evaluate a wide variety of architectural solutions to find the architecture that optimally meets the high-level requirements. Traditionally, these architectures and trades focus on product characteristics.

After the program system architecture solidifies, the system engineers segment the architecture into the various pieces for the IPTs, defining the requirements for the IPTs and the interfaces between the major architecture elements. A series of iterations rationalizes the requirements against the current best design solutions.

For example, the Teledesic functional requirement for "fiber-like" performance would be decomposed into quantitative requirements for latency, error rate, service availability and data throughput. Once this decomposition is complete, various system architectures are created and evaluated. This list of architectures could include geosynchronous satellites, Medium Earth Orbit (MEO) satellites, Low Earth Orbit (LEO) satellites, balloons, high-altitude planes and very tall microwave towers. [14] These architectures are traded against each other in terms of the decomposed network requirements. Through these trades, the optimal system architecture emerges: approximately 300 LEO satellites serving as routers-in-the-sky for a worldwide packet-switched network. [11]

Next, the subsystems of this architecture are identified, including the satellite, the user equipment and the satellite control center. [11] The requirements for these subsystems are defined, linking them to the program-level requirements through the system architecture. Also, the requirements for the interfaces between the various subsystems are defined and documented.

Once the system architecture and the interfaces are reasonably stable, the IPTs take over, creating a design to meet their functional requirements. This leads to a further decomposition of the design, where lower level IPTs create designs to meet the requirements defined by the higher-level IPT. This process could continue for several layers of decomposition, until the design can be implemented with commodity parts or by a small team of developers. For instance, the satellite subsystem is assigned to the Satellite IPT, which would define the product characteristics that would meet the satellite requirements. These requirements are product characteristics, such as weight, cost and size and the data communication protocols between the satellite and the user equipment, satellite and satellite control center and the satellite to itself. This product

architecture would divide the satellite into its major subsections, such as propulsion, avionics, payload and energy management. Requirements and interfaces would be captured for this lower level of decomposition.

This decomposition would proceed, taking the avionics subsection and breaking it into the elements that will accomplish the its requirements and interfaces. These elements could include a star camera, which determines the satellite orientation. At the level of the star camera, the system is decomposed enough to have the production centers produce a product meeting the star camera requirements. Once all the pieces of the satellite subsystems are at this level of decomposition, then the IPT design task is done.

While the actual process is much more iterative than this clear, linear flow, the flavor of the interactions, and the general top-down rationalization of the architecture in light of the lower level design solutions is a fair abstraction of the process. The rationalization of the requirements to the design is the core of Boeing's product system engineering methodology and deserves further explanation.

Rationalization of Requirements and Design

Often, tension exists between the requirements and the design. This tension can arise because two or more requirements are contradictory or mutually exclusive, the completion metrics are missing or poorly defined, or the requirements do not contribute to the completion of a program goal. Also, the requirements may be well structured, but overly constrain the design. A tension between the system engineers and the designers is inevitable on a project as risky and challenging as the Teledesic project.

The successful management of a complex project requires established procedures that create good requirements from the start and resolve any tensions that may arise. Boeing's methodology requires that the designers accept the requirement set for their subsystems. This puts the onus of communication on the system engineers and drives the requirements to be rationalized in light of the current design solution space. This rationalization occurs through several different types of interactions between the system engineers and the designers. The process is described as an interaction between the system engineering team and the

IPTs, although the process is equally valid for a higher-level IPT defining requirements for a lower-level IPT.

Contradictory Requirement Set

If the IPTs believe that their requirement set is inherently contradictory, then the system engineers must produce the trade that rationalized these requirements in terms of program-level outcomes. This process encourages the system engineers to quantitatively trade the requirements as they are completing their decompositions.

Weak Requirement Set

If the IPTs believe that the heritage of the requirement to program level goals is weak, then the system engineers must justify their initial linkage of the requirement back to a program level parent. This process encourages the system engineers to provide strong parent-child linkages between the requirement decomposition levels, eliminating unnecessary requirements at lower levels of decomposition.

Unverifiable Requirement Set

If the IPTs believe that the requirement is not verifiable, then the system engineers must justify the proposed verification metrics, methods, locations and schedule. This process encourages the system engineers to define clearly and quantitatively the success metrics for the IPTs and production centers. It also allows the project leadership to correctly schedule and cost the verification task early in the design process.

Difficult Requirement Set

If the IPTs understand the requirement set, but feel they cannot fulfill it with their current design, then the IPTs must produce a trade or analysis of their best alternatives, and quantify their deviation from the requirement. In this case, the IPTs must convince the system engineers of their analysis. Once this analysis has been accepted, the system engineers will trade the two requirements in terms of program goals and modify the less expensive requirement.

For instance, the satellite weight has strong heritage to the program cost through the launch cost. The program cost is also strongly dependent on the expected life of a satellite, which drives an on-orbit reliability requirement. If the best design has a failure rate below the reliability requirement, then redundant systems must be included. These systems add excess weight, which could cause the weight requirements to be out of specification. If the designers can illustrate that the requirements cannot be reconciled, then the on-orbit reliability requirement would be traded against the weight requirement, in terms of overall program cost. The requirement that was the least costly would be increased to accommodate the best current design.

The requirement-centric product system engineering methodology proves to be a robust way to define requirements and rationale them in light of the current design space. While iterative and seemingly full of conflict, the result is well-worded, complete, minimal, verifiable and rationalized product requirements.

3.2 Traditional Process System Engineering Methodology

In contrast, traditional process system engineering is centered on plans. This process begins by capturing the program level cost and schedule in the Program SOW, as outlined in Figure 8. Cost and schedule targets are then budgeted by project management to the various IPTs and production functions.

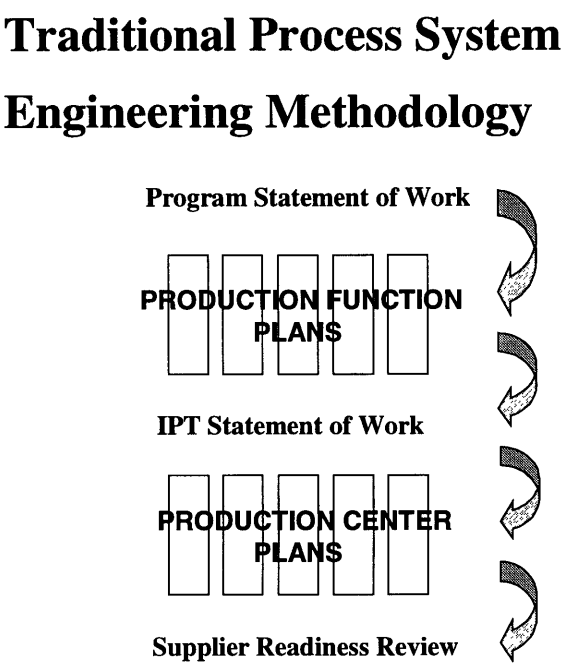

Figure 8: Traditional Process System Engineering Methodology

Based on the combination of the preliminary architecture designs, target costs and target schedule, the production functions create a plan that defines the functional strategies and processes to be deployed across the entire production system. These plans are referenced in the SOWs created by the IPT manufacturing and quality engineers. The IPT SOW is the heart of the production center execution contract with Boeing.

The production centers must present a plan that shows how they are going to meet the requirements of the IPT SOW and the program functional plans. As the project nears production, the IPT and the production functions complete on-site readiness reviews to ensure the production centers are executing their plans in a timely manner. The combination the production function plans, IPT SOWs and the production center plans make up the traditional process architecture documentation.

For instance, the process system design begins with the program cost and schedule requirements. The schedule requirement is decomposed from 5 years into the various stage requirements, such as system design in 1 year, detailed design and verification in 2 years and production and launch in 2 years. The cost requirement is allocated to the various segments, including the satellite segment. In this way, the cost and schedule are decomposed down to the lowest level process elements.

Once cost and schedule are decomposed, the production functions define their plans. Materiel, for instance, is responsible for procuring parts that meet the detailed design requirements at the allocated cost. They describe how the suppliers are selected, and the strategy if a supplier fails to meet the contract. Logistics describes the transportation methods and schedule between the production centers and ensures that costs and schedules are correctly counted in the component cost and schedule performance metrics. They also determine the spare strategy and ensure that the facilities for spare maintenance and storage are planned and budgeted. Manufacturing outlines the best processes for the required part geometry and determines the production system inventory control system. They also estimate the system flow time and maximum throughput. Quality determines the methods for production center product verification, estimates quality personnel costs and ensures verification time is counted in the overall process cycle time and capacity. System test determines the system verification processes, both on the ground and after the satellite is deployed into its orbit. DDE defines the system for communicating the data formats, interfaces and other relevant design information between the project and the production centers. Lastly, PM&P determines commodity part purchasing and quality verification.

Because the IPTs have the task of designing the product and process for their individual components, the application of the production function plans to the IPT SOWs is a source of tension. Usually this process is negotiated between the IPTs and the production functions, with exceptions to the production function plans captured in the IPT SOWs.

Because of the superiority of the product system engineering methodology, product designers usually win negotiations with the manufacturing engineers in the IPT design process. For instance, the manufacturing plan may call for creating structural holes in one cutting operation, in order to eliminate a reaming operation that adds cost, cycle time and clean room contamination. However, the tolerance on the hole may be tighter than the cutting process capability, requiring the reaming operation. Product designers can justify their tolerance as necessary to achieve a structural rigidity requirement. Because the rigidity requirement has strong lineage to program-level goals, detailed trade studies and quantified verification metrics, the tolerance requirement stays and the manufacturing plan requirement is exempted in the product's SOW. This dynamic leaves the manufacturing engineers lacking the necessary facts and data to effectively defend the key elements of the process and supply chain design, even through they have full membership in the IPT design process.

Once the IPT SOWs have incorporated all the production function negotiations, the production centers create plans that demonstrate their ability to implement the IPT SOW. Often, the IPT SOW is sub-optimal for the established methods in a particular plant. In this case, the production center would negotiate with the IPT and the relevant production function to modify the requirements by demonstrating lower cost or higher quality. For instance, if the IPT SOW specifies inspection of a key product characteristic that the plant verifies through statistical process control, then the IPT and the Quality function would agree to accept the process control data for the entire production run instead of measurement data for every part. Like the production plan exemptions, IPT SOW changes are often a result of losing trades with the stronger functional requirements.

As full production approaches, the IPTs and production functions conduct a readiness review to ensure that the production center is implementing its plan. Any irregularities are fixed through corrective action plans. In rare cases, the supplier fails the review and drastic measures are taken. These measures vary from Boeing lending the production center engineering "support" to moving the product to a new factory.

3.3 Traditional Supply Chain System Engineering Methodology

Traditionally, Boeing does not explicitly engineer the supply chain architecture. The make/buy strategy and the selection of strategic suppliers resides with the project management, who choose the make/buy content based on trades between internal production availability, strategic technology investment and program risk. After the project management selects the strategic "cherished partners", the rest of the production centers are selected by the IPTs. External suppliers are typically chosen and managed through a multi-supplier, competitive bid arrangement. As outlined above, the coordination of the logistics and data flow between the suppliers were managed through the functional plans and flowed to the suppliers through the IPT SOWs.

Once the suppliers are selected, they are managed through the activities of several of the production functions. Quality evaluates supplier performance. Materiel manages the supplier legal relationship, determining the project sourcing strategy and negotiating product price. Logistics coordinates the transportation, maintenance and repair activities. DDE supports the communication of drawings, change requests and requirement specifications between the project and the suppliers. Manufacturing supports process readiness reviews, inventory holding requirements and production system bottleneck identification and mitigation activities. System Test verifies final assembly, on-orbit performance and supplier design qualification. PM&P manages the quality and price of the commodity part suppliers.

In summary, the traditional system engineering design methodology reveals two distinct processes, a requirement-centric product methodology and a plan-centric process methodology. The supply chain architecture is implicitly designed through the process architecture activities. The quantitative nature and rigorous rationalization of the product methodology makes the product requirement rationale superior to the process and supply chain requirement rationale. This superior rationale enables the product design to dominate the process and supply chain design in the IPT design process, leading to a consistent bias toward the product design despite the strong IPT concurrent engineering emphasis.

4 Proposed System Engineering Methodology Expansion

To enable the product, process and supply chain architectures to be equally optimized, the system engineering methodology must be common across all architectures. Without this common methodology, it is impossible to have a common language between the product and process system engineers, deliver a consistent requirement package to the IPTs and production centers or trade aspects of these architectures on equal footing. So, reconciliation of the traditionally dissimilar methodologies is the first task.

The goal is to expand the requirement-centric product system engineering methodology into the process and supply chain architectures, as outlined in Figure 9. This expansion is achieved by taking the traditional plans and translating them into a requirement hierarchy. Once this requirement hierarchy is completed and rationalized, then the requirements can be extracted back into the traditional plan form, if desired. The detailed process for this translation is outlined below. This expansion of the product system engineering methodology into the process and supply chain architectures achieves the goal of a common development process that is able to trade requirements across product, process and supply chain architectures.

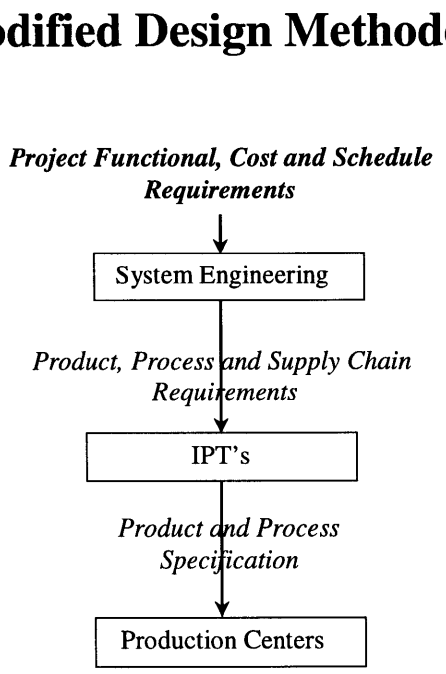

Figure 9: Expanded System Engineering Methodology

Another feature of the system engineering expansion is to define the supply chain architecture independently from the process architecture. The supply chain architecture explicitly declares the supplier selection strategy, the make/buy strategy, the logistics strategy, the operational strategy and the supplier communication process. Also, the supply chain requirements are independently defined from the process requirements. This segmentation of the supply chain architecture allows for the explicit design of the supply chain and the consistent application of that architecture across all the production centers.

4.1 Advantages of Proposed Methodology Expansion

The requirement-centric product methodology proves to be superior to the plan-centric process methodology. The quantitative nature of the requirement definition process, clear linkages back to the program goals and defined processes for the rationalization of the requirements to the design make the expansion of the product methodology into the process and supply chain architectures the best way to reconcile the two processes. The advantages of the expanded system engineering methodology over the traditional methodology are outlined below.

Quantitative Trades Across Architecture Boundaries

Because of the quantitative nature of the requirement definition and the clear linkage back to the high-level program requirements, low-level requirements to product, process and supply chain architectures can be quickly evaluated and traded in terms of program-level goals. The process and supply chain requirements will be quantified and linked like their product requirement brethren, enabling the project to trade requirements across architecture boundaries with a common methodology. The trade of the process flow time versus the satellite weight outlined below is an excellent example of this ability to trade across architecture boundaries.

Complete Decomposition of Project Requirements

Because of the clear linkage between the various levels of decomposition, requirements can be consistently evaluated across architecture boundaries. The set of requirements at the lowest levels of the product, process and supply chain architectures can be extracted and evaluated to ensure that they completely define the necessary requirements to meet the overall project goals.

For instance, the overall supply chain flow time, from initial component acquisition to satellite orbit insertion, can be extracted from each of the IPT process requirements. This definition of the overall flow time enables the program to quantify the cost of repair late in the production flow, in addition to knowing the cost of the inventory of the entire production system. This data will help to correctly trade the costs of repairing or scraping a damaged component. Also, each segment's contribution to the overall system flow time can be quantified, helping to direct improvement efforts to the segments where improvement would contribute the most to the program goals.

Minimal Decomposition of Project Requirements

The clear linkage between the high and low level requirements and the defined requirement rationalization processes also allows for the elimination of duplicate requirements and the modification of contradictory requirements within the process and supply chain architectures. This ability to achieve minimal decomposition is almost impossible in the plan-centric process, due to the difficulty of comprehending every part of every plan concurrently.

In the same way designers have a natural tendency to "gold plate" their designs, production functions tend to locally add elements to the process or supply chain architecture that they feel are important. For instance, one materiel manager's negative experience with a supplier may trigger the specification of extra data requirements, increasing the overall program cost. The explicit check for feature creep in the process and supply chain requirements will minimize the overall cost to the project.

Quantitative Performance Evaluation

The requirement metrics become binding success criteria for both the IPTs and the production centers. The measurable, quantifiable nature of the requirement metrics makes them an ideal communication tool for defining the expectations with production centers in clear and unambiguous terms.

Instead of reviewing the production center plan of the SOW exemptions of the functional plans, the requirement-centric review process is conducted with measurable verification metrics established early in the design cycle. Conformance is clearly determined and the deviation from the requirement can be quantified. The linkage between the system requirements and the production readiness evaluation is clear and direct, allowing for a fast and detailed evaluation of project risk. This evaluation of the supplier deviation in terms of overall project risk is much more difficult in the traditional plan-centric methodology.

Program Feasibility Assessment

The rationalization of the requirements to the design is an important mechanism for determining program feasibility. Agreement between the system engineers and the IPTs is a credible signal to management that the project is feasible. Extending this program feasibility visibility to the process and supply chain requirements allows the management team to accurately assess the overall program risk at significant project milestones.

Organizational Learning and Core Competence Development

The documentation of the design rationalization process in the form of trades also provides tangible learning tools. The clear decision path and the quantitative nature of the analysis make learning from project to project much easier to communicate, as well as facilitating the rapid initiation of new team members to the past decisions that frame their current work. Especially since the satellite production and supply chain systems are new for the industry, this learning could grow to be a significant source of competitive advantage for Boeing.

As these points illustrate, the requirement-centric methodology has a host of attractive benefits that make it the methodology of choice. The quantitative nature of the requirement definition process, clear linkages back to the program goals and defined processes for the rationalization of the requirements to the design will greatly improve the traditional plan-centric process system engineering methodology. This improvement makes the transition to the requirement-centric process an important priority for the Teledesic project leadership.

4.2 The Methodology Expansion Strategy

The recommendation to expand the requirement-centric product system engineering process into the process and supply chain architectures is made in the midst of the ongoing system engineering effort. Luckily, the creation of the DefineProduce team within the System Engineering team enables the ability to define the process and supply chain system engineering activities. However, the production functions, while reporting to the project through the DefineProduce team, have a strong heritage in the plan-centric methodology. Since valuable work was accomplished in plan form, the implementation strategy allows for the initial development of the requirements in plan form and maintains the ability to view the requirements as a plan at the end of the process.

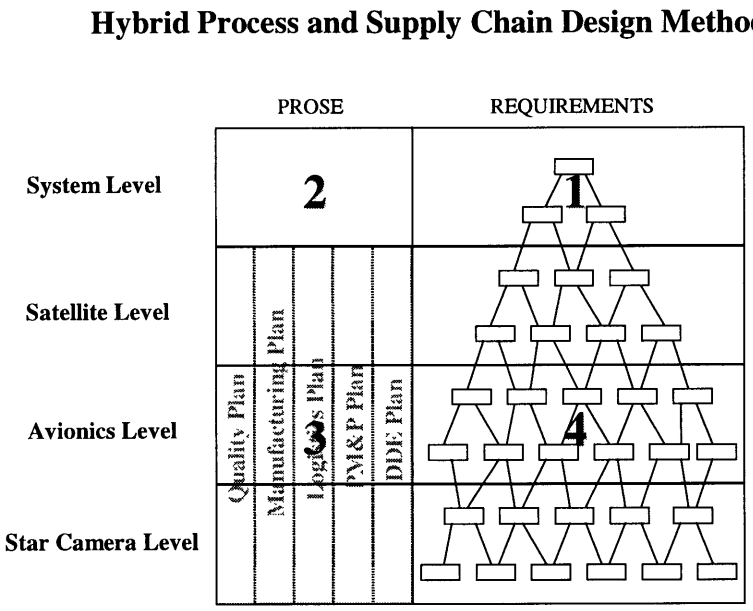

Figure 10: Hybrid Process and Supply Chain System Engineering Process

The transition from the plan view to the requirement view back to the plan view was executed in four major steps, following the numbers in Figure 10.

Step 1

The requirements that drive the process and supply chain architectures are identified and put into requirement form, with verification metrics and schedules. These requirements became the very top boxes of the process and supply chain requirement hierarchies.

Step 2

A program-level production and supply chain architecture is created, integrating into one document all the production function strategies for achieving the program process and supply chain requirements.

Step 3

The production function managers make traditional functional plans.

Step 4

Starting with the program plan and iterating through each architecture level, the plans are translated into full, complete requirement statements through the following process:

1. The plan is parsed into statements.
2. The statements are re-written in requirement-like language.
3. Confusing requirements are broken into multiple requirements.
4. Redundant requirements are eliminated.
5. The requirements are linked so that every requirement has a higher level parent.
6. The requirements are made product, process or supply chain specific.
7. Verification statements are added to all the requirements, specifying the metrics, methods, locations and times of the verification activities.
8. The IPT that will implement the requirement is identified.
9. IPT agreement to the requirement set is negotiated.
10. The ownership of the requirement is transferred to the IPTs.
11. The plans are recreated by merging the requirements and verification statements into paragraph form.

12. Descriptive statements are added to the requirement paragraphs to contextualize the requirement-like prose.

These four steps fully translate the plans into requirements, complete, link and rationalize the requirements, obtain IPT acceptance of the requirements and then re-package the requirements into a plan view. This hybrid process, while more effort than pure requirement decomposition, allows the production functions to develop their strategies in a comfortable format. For the IPT product designers, the ability to see production and supply chain requirements in the same form as the traditional product requirements gives the new requirements more credibility than the traditional DFM guidelines and manufacturing "lore". Well-phrased requirement statements, clearly defined success criteria, explicit linkage to program-level requirements and quantitative trade studies put the process and supply chain requirements on equal footing with the product requirements.

5 The Expanded Methodology in Action

The expansion of the requirement-centric product system engineering methodology to the process and supply chain architectures has been outlined, including the advantages and implementation strategy. So, what were the results? Two examples from the Teledesic project illustrate the effectiveness of the expanded methodology in optimizing across architecture boundaries.

5.1 The Struggle between Flow Time and Weight

The first real test of the expanded system engineering methodology occurred over how "modular" the Teledesic design should be. One of the difficult production requirements is short material flow time. By shortening the flow time for all assemblies, the costs of the work in process is reduced. Also, because the satellite production must be completed before the system can be operational, less flow time directly translates into earlier project completion. Therefore, production flow time strongly contributes to the overall program cost and schedule. Because the launch dates were beginning to be negotiated, determination of the production flow time needed to be solidified. The strategy for reducing the overall material flow was to divide up the satellite into several large pieces and produce those pieces in parallel. This strategy lead to a requirement for a modular design: a satellite that could be physically broken up and assembled in parallel.

The launch cost target was also set very aggressively. In order to meet this cost, many satellites where planned to be deployed in one launch vehicle, putting a strong restriction on the satellite weight and volume. The initial reconciliation of the satellite design to the satellite requirements showed that the greatest deviation between the design and the specification was the satellite weight. This deviation lead to a strong re-design effort to optimize for weight. Because joints require excess material, the baseline design went from several to two modules. The design community said that they had met the modular requirement, because the satellite could, indeed, be divided into modules and assembled in parallel, although with a much longer flow time than with the original design.

In the traditional process, this disagreement would have been problematic for the IPT manufacturing engineers. Because the goal for a modular design would have been captured in the manufacturing plan and not captured in the requirements document, the decision to compromise modularity for weight might have happened unconsciously. Even if the conflict were recognized, the weight requirement would have dominated due to its more rigorous specification and rationalization.

In the expanded methodology, however, both the modular design requirement and the weight requirement are linked back to program cost, as shown in Figure 11. Therefore, they can be traded against each other by quantifying what a pound of weight is worth and what a day of flow time is worth. The DefineProduce team evaluated the cost of a day of flow time by measuring the production cost savings for the project if the production run is shortened by one day. The product system engineers evaluated the cost of building, launching and deploying a pound of satellite. From this analysis, the IPT was able to optimize the design across product and process boundaries, in terms of overall project cost. Without defining the process requirements in the same manner and with the same level of quantification as the product requirements, this trade could not be completed in a structured and analytical way.

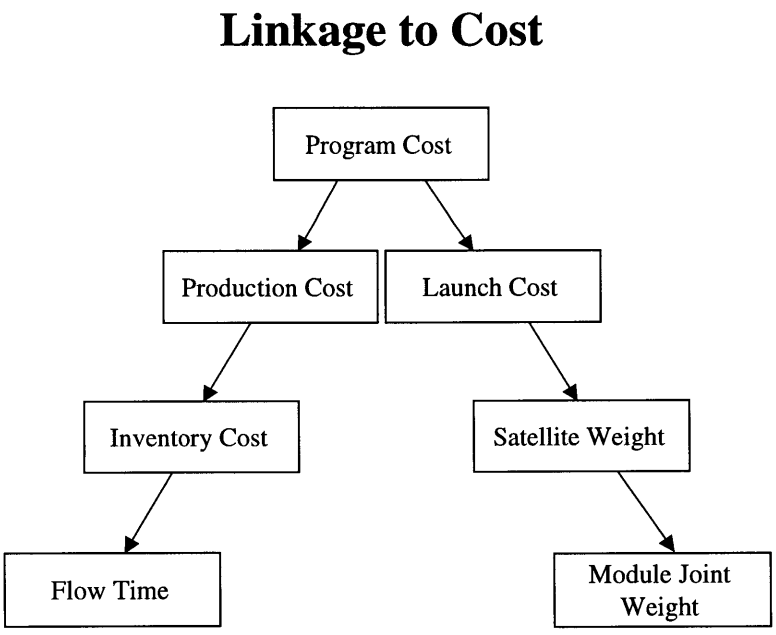

Figure 11: Weight and Modular Design Linkage to Cost

5.2 Performance Verification Mental Models Clash

A second major test of the expanded methodology grew out of a design verification debate. System test for the satellite is very expensive and time-consuming. A strategy for reducing the system test cost and flow time was to push the product performance verification to the lowest possible level of assembly. This requirement enables faulty components to be quickly identified and fixed, as well as reduces the complexity of the test equipment in final assembly. After the basic assemblies are tested, only the assembled interfaces are verified at the later stages of production.

Because the product design community traditionally thinks in term of functional decomposition, they call a design "modular" if all the elements that make up a functional subsystem are located on the same piece of structure. For instance, a good modular design would keep the star camera grouped with the other avionics equipment, the thrusters grouped with the rest of the propulsion equipment and the communication antennas kept with the payload. Because the functional subsystems have verification metrics, they interpreted the combination of the modular assembly requirement and the verification requirement to mean that the avionics suite has to be co-located on one piece of structure so that the avionics can be tested in the subassembly factory. Because weight constraints made it impossible to keep all functional elements on separate structures, the designers declared the early testing requirement infeasible.

The DefineProduce team, on the other extreme, was strongly supporting a robust design approach, where the design requirements would be much narrower than the process capability, eliminating the need for testing altogether. However, they knew that the process capability for many new processes couldn't be quantified, due to a lack of data. For these cases, the requirement for testing at the lowest level of assembly was seen as the next best option. With this strategy, once the design is tested and verified, only performance at the lowest levels of assembly needs to be tested. If these parts are within specification and the higher level of assembly processes are completed within tolerances, then the avionics, propulsion, payload or power systems shouldn't require functional verification as a suite.

Through the rationalization process for the verification strategy requirements, the mental model differences between the two groups surfaced. The design community was not convinced that the low-level test strategy was superior to the traditional final assembly test strategy. The DefineProduce team was able to show that the savings of a shorter cycle and repair rework time over the life of the production run was much greater than the increased testing of the design qualification units. The conclusion was the acceptance of the low-level test requirement. Also, modular design was defined as a satellite that could be physically broken up and assembled in parallel. Lastly, the design community accepted the strategy of not testing the functional suites in full production.

Because historically the product requirements drove the production verification processes, the product design assumptions of system verification normally would have significantly increased the final assembly cost and flow time. However, because of the decomposition of the production system costs, the inefficiencies of the old system were able to measured and the superior strategy could be quantitatively justified. By being able to trade across product and process boundaries, the costs of assumptions in the product architecture were accurately measured in the process architecture. In this case, the solution greatly reduces production system cost without any negative impacts to the product functionality.

These two examples illustrate how the expanded methodology allows for the optimization of requirements across architecture boundaries. This ability to reconcile historically dissimilar system engineering methodologies enables a richer dialogue between functional groups which, in turn, produces better product, process and supply chain architectures.

6 Conclusion

This work attempts to improve Boeing's world-class system engineering competency by expanding the superior product system engineering methodology into the process and supply chain architectures. This expansion integrates the product, process and supply chain architectures in the concept phase, where much of the production and supply chain system costs are determined. Current industry best practice, like Boeing's traditional process, executes concurrent engineering at the IPT design stage. This expansion of one of Boeing's core competencies could be a key competitive advantage in the race to be the global provider of satellite systems. In defining and implementing the methodology expansion, several conclusions became clear.

The first insight is that a consistent methodology across the product, process and supply chain design activities is key to optimizing overall program goals. Both the modularity and verification trades illustrate the opportunity for global optimization through analysis across traditional boundaries. Without a common methodology, much of the leverage of adding process and supply chain system engineering to the concept phase will be lost due to different languages and mental models conspiring to achieve traditional outcomes. Only by forcing a common process, a common language and a common set of tools can true concurrent system engineering of the product, process and supply chain architectures occur.

Second, requirements are superior to plans in specifying and defining system architectures. Clear and specific language, quantifiable verification metrics, explicit linkages to program-level goals and explicit refinement and acceptance methods make a requirement-centric process vastly superior to a plan-centric process. Requirements are necessary to optimize the design across architecture boundaries.

Third, challenging program goals and strategic organizational changes are key enablers to any process improvement effort. The challenge of creating one satellite per day, when the industry best system produces one satellite per week [8] causes senior satellite designers to be open to new ideas. In addition, the inclusion of the DefineProduce team in the concept phase of the project shows senior management

commitment to doing system engineering in a new way. By having a voice at the very beginning of the project, the production system can truly impact the product design. Significant risk in the process and supply chain architectures and strong organizational positioning were key enablers of the creation of a process and supply chain system engineering methodology and implementing it within a strong product design tradition.

6.1 Further Research Opportunities

In the course of completing this work, several promising opportunities for further research surfaced. The ability to estimate process capability without having the actual equipment would help focus production system design efforts. Also, the segmentation of the supply chain architecture surfaced a conflict between outsourcing to reduce program risk and making the product in-house to build future competencies. Lastly, production system engineering might be able to leverage current advances in software system engineering in increase production system design re-use and minimize unnecessary production design costs.

Process Capability Estimation Tools

The best tools for establishing the relationship between the product design and process capability; Robust Design, DFX and Key Characteristics (KC) are based on existing process capability. However, a key element of any process design activity is to quantify and measure the program risk in the creation and implementation of new process capabilities. The ability to predict the risk and cost of closing the performance gap is fundamental to assessing overall project risk. Many of the problems in the Teledesic production system will be in ramping up these new processes to their necessary performance levels. Currently, no good tools exist for estimating which of these processes will be problematic and should, therefore, be mitigated first. Analytic tools to assist in this assessment would greatly aid the process architecture effort.

Optimization of Organizational and Program Goals

The requirement-centric methodology assumes that the optimal achievement of the program level goals is in the design organization's best interests. However, optimal product architectures often leave the design organization with low value-added pieces of the design. [9] A system to quantitatively trade the risk of

project failure against the risk of company competence obsolescence would be a valuable extension to the traditional project focused methodology.

"Object Oriented" Production System Engineering

Recent changes in the design and implementation of large software systems may lend a helping hand in addressing the lack of good new production center design tools. In the last few years, software methodologies have shifted from a functional or data flow topology to an "object-oriented" topology. The result is a clearly defined interface between the raw building blocks of software code, centered on objects that are more likely to be re-useable across product platforms. For instance, most software programs have a user interface, a communication module, a computational module, a database and perhaps an image generation module. By centering the design on these relatively constant objects, re-use across projects is substantially increased over a functional architecture. This re-use is achieved because a product family's "look and feel" is defined by consistent object implementation, but the products within the family are segmented by feature set. A methodology based on objects, therefore, is more robust over several projects.

Architecture design in this framework becomes a matter of investigating the currently available objects and relating them in such a way as to optimally meet the requirement set. Only if the current set of objects is unable to meet the new requirement set are new objects created. In this new endeavor, old objects can be extended or modified to meet the new requirement set, or new objects can be created from scratch. In either case, the clear interface requirements and the quantitative deviation from the required functionality drive the object design to completely but minimally satisfy the project requirement. This new object then joins the library of objects available for future use.

Substituting production processes for objects, the production architecture design task could amount to the arrangement of production processes within production centers into logical relationships. Only in the case where the present capabilities could not fulfill the program requirements would investment in improved or new processes be undertaken. The clear requirement and delta from its achievement would drive the improvement to completely but minimally satisfy the project requirement.

Like in software, the tasks of production center performance and architecture design effectiveness would be neatly segmented into two tasks with clearly defined metrics. Instead of a mish-mash of DFX guidelines, robust design parameters and quality metrics to guide the design task, clear, uniform process capability metrics against standard design features would serve as the communication medium between production and design. This clear interface would give internal production center managers clear signals to optimize or improve production capability, as well as quantify and measure the variety and complexity of the design feature set.

Endnotes

[1] Cleland, David (1996) Chapter 10: Concurrent Engineering Teams, Strategic Management of Teams, 198-215, John Wiley and Sons, Inc.

[2] Clark, Kim and Wheelwright, Steven, (1992) Managing New Product and Process Development, Free Press

[3] Womack, James P., et. al. (1991) The Machine that Changed the World, HarperCollins Publishers

[4] Shiba, Shoji, et. al. (1993) A New American TQM, Productivity Press

[5] Goldratt, Eliyahu (1984) The Goal, North River Press

[6] Winston, Wayne (1995) Introduction to Mathematical Programming, Second Edition, Duxbury Press

[7] Mlynarczyk, Mlynar (1995) Improving Synergy in Multi-Site Microprocessor Manufacturing: An Analysis of a Copy Exactly Approach, LFM Master's Thesis

[8] Haines, Thomas (12-7-97) *Teledesic: a dare on deadline,* Seattle Times, A1, A5-6

[9] Fine, Charles (Forthcoming) *Clockspeed: Winning Industry Control in the Age of Temporary Advantage*, Perseus Books, Chapter 7

[10] Crawley, Edward (Fall 1998) Total Holistic View of Product/Process Architecture, Class Overhead, System Architecture Design, MIT

[11] http://www.teledesic.com/overview/fastfact.html

[12] (9-29-93) *Iridium, INC. Holds First Board Of Directors Meeting —Corporate Officers Elected,* Iridium Press Release

[13] Nye, Sheridan (12-18-1997) *Russian Satellite Offer Flies Past Hughes*, Total Telecom News Service

[14] Pelton, Joseph, *Telecommunications in the 21st Century*, Scientific American, April 1998, pp.80-85.

[15] Teledesic FCC proposal

References

Clark, Kim and Wheelwright, Steven, (1992) Managing New Product and Process Development, Free Press

Clark, Kim and Fujimoto, Takahiro (1991) Product Development Performance : Strategy, Organization, and Management in the World Auto Industry, Harvard Business School Press

Ulrich, Karl, (1995) *The role of product architecture in the manufacturing firm*, Research Policy Vol. 24, p.419-440.

Crawley, Edward (Fall 1998) Total Holistic View of Product/Process Architecture, Class Overhead, System Architecture Design, MIT

Fine, Charles (Forthcoming) *Clockspeed: Winning Industry Control in the Age of Temporary Advantage*, Perseus Books

Fine, Charles and Whitney, Daniel, (2-96) *Is the Make-Buy Decision Process a Core Competence?* , MIT Working Paper